虚拟机类加载机制

2022年1月19日 11:41

类加载时机.

顺序不固定.

解析可以在初始化后支持Java动态绑定.

图7-1 类的生命周期

类加载过程.

①加载.

1）通过一个类的全限定名来获取定义此类的二进制字节流。→ 加载到方法区.

2）将这个字节流所代表的静态存储结构转化为方法区的运行时数据结构。

3）在内存中生成一个代表这个类的java.lang.Class对象，作为方法区这个类的各种数据的访问入口。

多种灵活加载方式.

·从ZIP压缩包中读取，这很常见，最终成为日后JAR、EAR、WAR格式的基础。

·从网络中获取，这种场景最典型的应用就是Web Applet。

·运行时计算生成，这种场景使用得最多的就是动态代理技术，在java.lang.reflect.Proxy中，就是用了ProxyGenerator.generateProxyClass()来为特定接口生成形式为"*$Proxy"的代理类的二进制字节流。

·由其他文件生成，典型场景是JSP应用，由JSP文件生成对应的Class文件。

·从数据库中读取，这种场景相对少见些，例如有些中间件服务器（如SAP Netweaver）可以选择把程序安装到数据库中来完成程序代码在集群间的分发。

·可以从加密文件中获取，这是典型的防Class文件被反编译的保护措施，通过加载时解密Class文件来保障程序运行逻辑不被窥探。

②验证. 确保class文件符合JVM规范.

保证不会对JVM有影响.

四个阶段.

(1) 文件格式验证.

(2) 元数据验证.

(3) 字节码验证.

(4) 符号引用验证.

③准备.

→ 非实例.

类变量内存分配.

初始零值.

类变量内存分配.

初始零值.

④解析.

符号引用→直接引用.

⑤初始化.

<clinit>() 由编译器合并 static{}、类变量赋值组成.

类加载器.

①类与类加载器.

不同 ClassLoader 加载的 class 文件用 instanceof 判断为 false.

②双亲委派模型.

① Bootstrap Class Loader 加载 %Java_home%/lib.

② Extension Class Loader 加载 %Java_home%/lib/ext.

③ Application Class Loader 加载用户类路径 classPath.

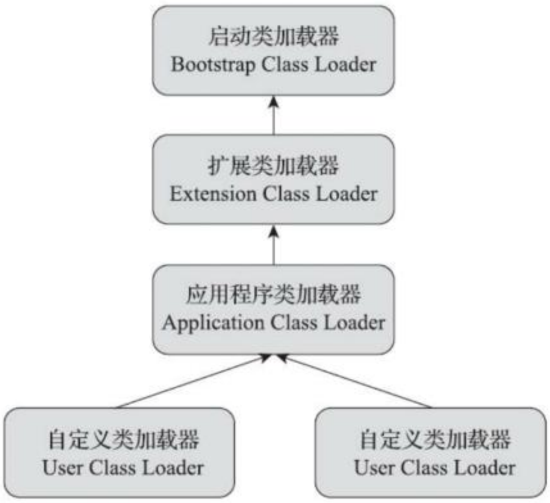

图7-2 类加载器双亲委派模型

(1) 工作过程.

一层层往上找，都找不到再加载.

(2) 优点.

越基础的类由越上层加载.

优先级的层次关系. Object对象由顶层加载.
保证各种加载环境为同一类.

③ OSGI对模型的"破坏".

↘ Hot Swap / Hot Deployment.

Java模块化系统. JPMS (Jigsaw)

Java模块化系统. JPMS. (Jdk9).

目标：模块化目标（可配置的封装隔离机制）.

①解决的问题.

(1) 避免大部分运行时类型依赖异常.

模块对其他模块的依赖显式声明.

(2) 跨Jar包 public 访问问题.

必须明确声明哪些 public 可访问.

②模块的兼容性. 兼容类路径查找机制.

提出 (ModulePath)

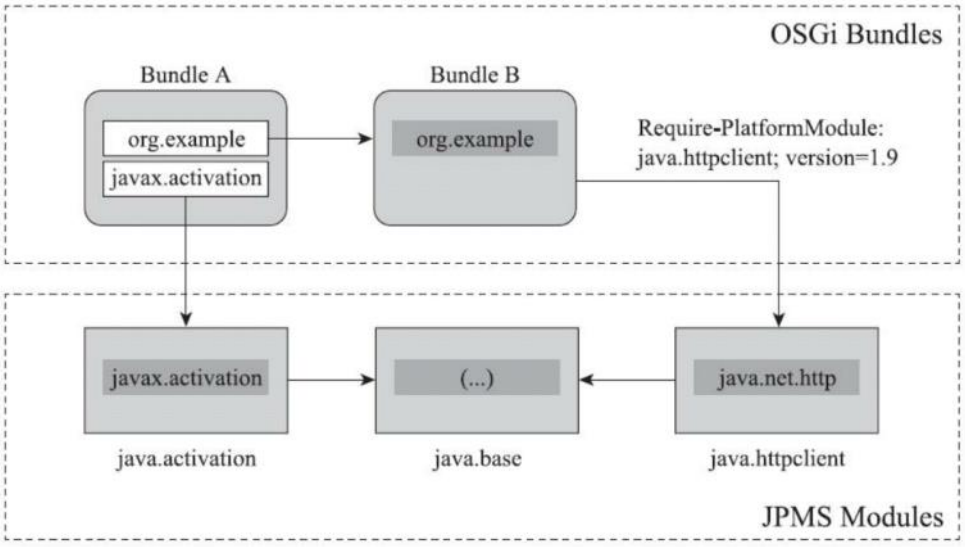

图7-4 OSGi与JPMS交互[4]

③模块化下的类加载器.

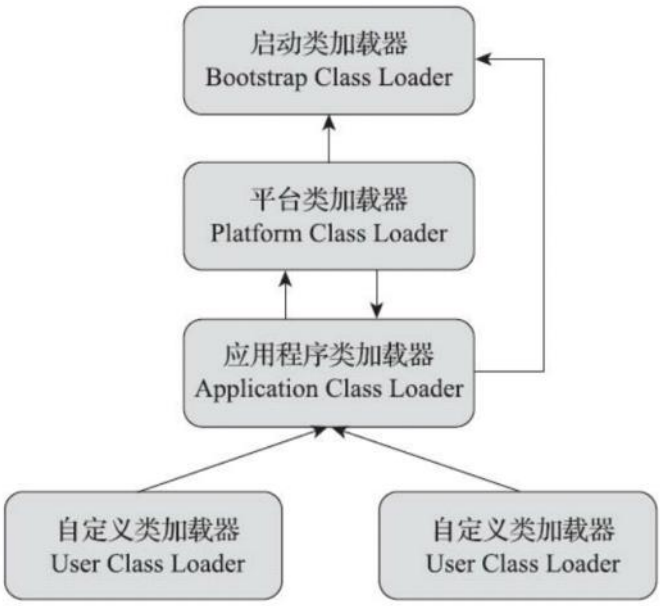

图7-7　JDK 9后的类加载器委派关系